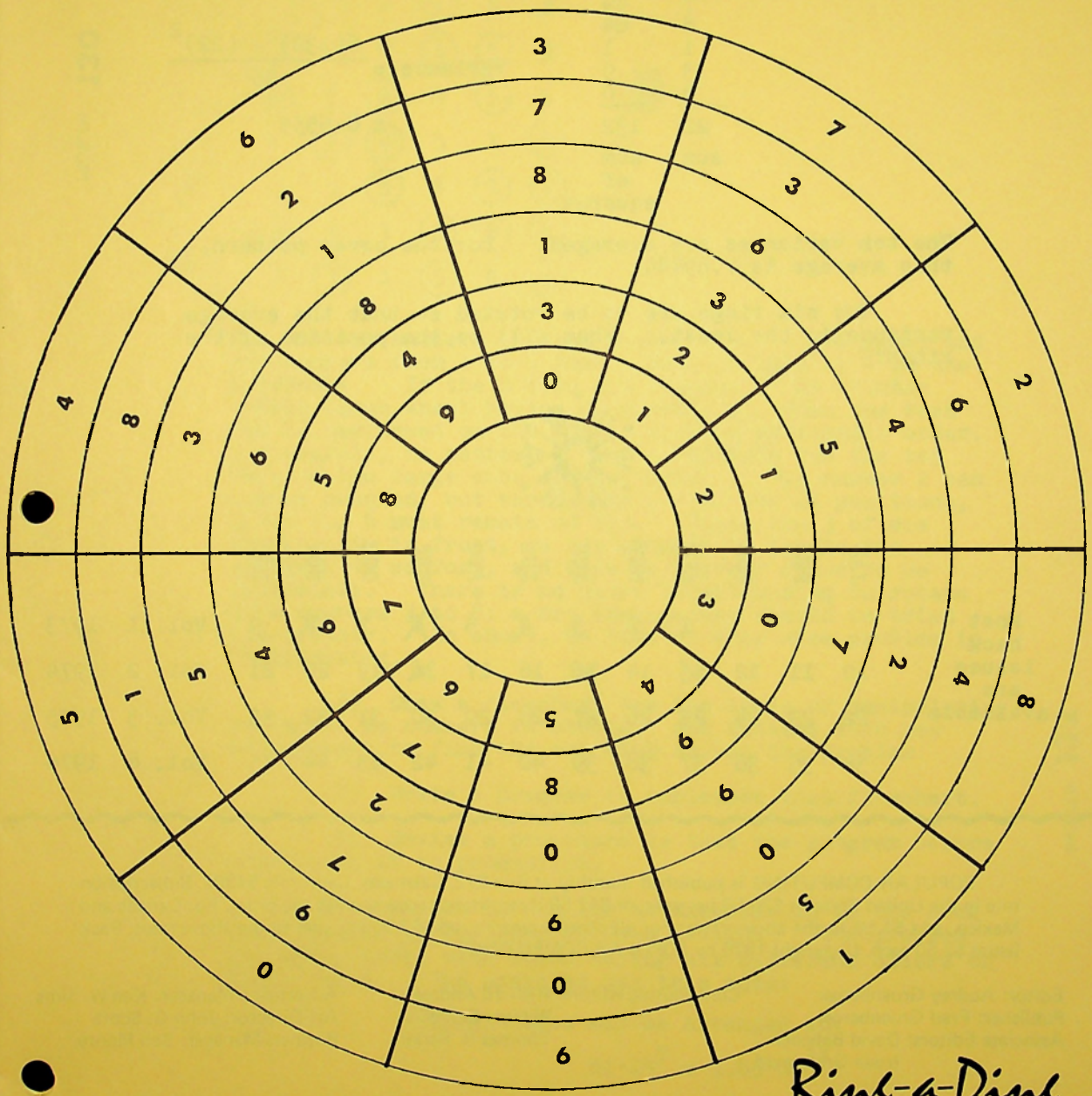
Popular Computing 39

June 1976 Volume 4 Number 6



Ring-a-Ding

In the diagram on the cover, the six rings each contain the digits from zero through 9 (to be read looking out from the center). For each of the ten sectors, the variance of the six numbers is calculated. For example, for the top sector:

3	9
7	49
8	64
1	1
3	9
0	0
22	132
sum	sum of squares

$$\text{Variance} = \frac{6(132) - (22)^2}{36}$$

$$= 8.5555$$

PROBLEM 130

The ten variances are averaged. For the cover pattern, this average is 7.69444.

The six rings are to be rotated so that the average variance is the least. What will be the position of the rings?

POPULAR COMPUTING is published monthly at Box 272, Calabasas, California 91302. Subscription rate in the United States is $20.50 per year, or $17.50 if remittance accompanies the order. For Canada and Mexico, add $1.50 to the above rates. For all other countries, add $3.50 per year to the above rates. Back issues $2.50 each. Copyright 1976 by POPULAR COMPUTING.

Editor: Audrey Gruenberger
Publisher: Fred Gruenberger
Associate Editors: David Babcock
Irwin Greenwald

Contributing editors: Richard Andree
William C. McGee
Thomas R. Parkin

Advertising Manager. Ken W. Sims
Art Director: John G. Scott
Business Manager: Ben Moore

Problem Solution

In issue No. 17, Problem 56 was the following:

Take the proper fractions in lowest terms in ascending order by denominators and within each denominator in ascending order by numerators (an ordering first used in issue No. 3 in the Number the Fractions Problem). Express each fraction in binary. Take the first bit of the first fraction, the second bit of the second fraction,..., the Kth bit of the Kth fraction, as follows:

1/2	.(1)0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1/3	. 0(1)0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
2/3	. 1 0(1)0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
1/4	. 0 1 0(0)0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
3/4	. 1 1 0 0(0)0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1/5	. 0 0 1 1 0(0)1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0
2/5	. 0 1 1 0 0 1(1)0 0 1 1 0 0 1 1 0 0 1 1 0 0 1
3/5	. 1 0 0 1 1 0 0(1)1 0 0 1 1 0 0 1 1 0 0 1 1 0
4/5	. 1 1 0 0 1 1 0 0(1)1 0 0 1 1 0 0 1 1 0 0 1 1
1/6	. 0 0 1 0 1 0 1 0 1(0)1 0 1 0 1 0 1 0 1 0 1 0
5/6	. 1 1 0 1 0 1 0 1 0 1(0)1 0 1 0 1 0 1 0 1 0 1
1/7	. 0 0 1 0 0 1 0 0 1 0 0(1)0 0 1 0 0 1 0 0 1 0
2/7	. 0 1 0 0 1 0 0 1 0 0 1 0(0)1 0 0 1 0 0 1 0 0
3/7	. 0 1 1 0 1 1 0 1 1 0 1 1 0(1)1 0 1 1 0 1 1 0

and these bits form a new (irrational) number. The <u>decimal</u> value of that number was sought.

Associate Editor David Babcock wrote an assembly language program to calculate the first 1000 bits of the required number and thus calculate the first 300 decimal digits, given here:

.88900 03549 69523 01393 46719 69087 91293 10538 48021 01795
43414 54346 53700 32396 35085 25312 86110 98458 18655 73176
99818 10555 46085 78199 67269 34108 50001 90379 54057 44168
77307 33917 94335 85278 35527 43778 57018 04188 56287 53592
09241 48928 57072 99181 37374 46856 95544 82361 08493 46707
04051 78204 45434 51338 68333 41601 78380 52871 19920 02911 □

CONTEST 8

OUTGUESS

Our 8th contest is a guessing game, with no computing problem involved (but the $25 prize is just as real as ever).

Send in as your entry a set of nine numbers, chosen as you please in the range between 100 and 999. For all the numbers received, we will calculate the mean and standard deviation, and form the product of those two values. For your entry, you calculate the mean and standard deviation of your set of nine numbers, and the product of the two values. The winner will be the entry for which the product comes closest to the product for all the numbers received. An entry form is given for your convenience.

Suppose, for example, that all the sets of nine numbers were uniformly distributed over the stated range. The mean would then be 500, and the standard deviation would be 275, giving a product of 137500. An entry, then, of a set of nine numbers with a mean of 500 and a standard deviation of 275 would be the winner. However, one entry with this set of numbers:

100, 100, 100, 100, 100, 100, 100, 100, 999

would have a mean of 199.88 and a standard deviation of 299.66 (with a product, then, of 59900) and that entry might significantly bias the mean and standard deviation of the total set of all numbers.

Similarly, a single entry of this set of numbers:

999, 999, 999, 999, 999, 999, 999, 999, 100

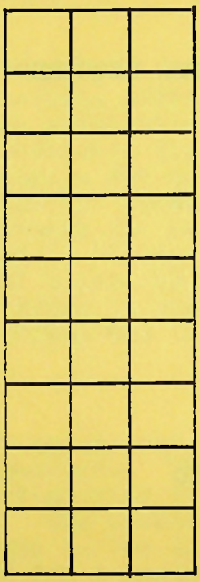
would yield values of 899.11, 299.66, and 270000 and might again seriously bias the totals.

Each person may submit <u>two</u> sets of nine numbers. The calculations for the means and standard deviations must be shown (and incorrect arithmetic will invalidate an entry). All entries must be received by POPULAR COMPUTING by August 31, 1976. This contest is just for fun, with no deep thinking required.

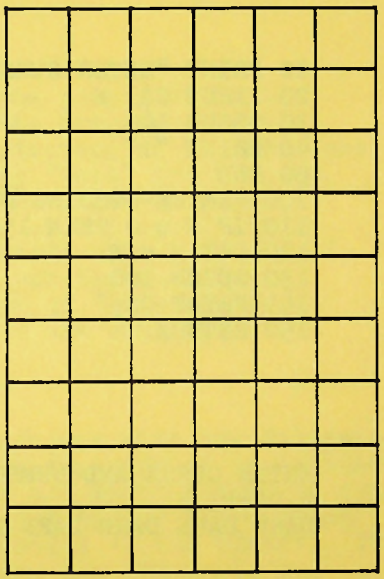
Entry Form

Popular Computing
BOX 272 CALABASAS, CA. 91302

Enter nine 3-digit numbers here →



and the squares of the nine numbers here →



PROBLEM 131

Sums:

Mean:

Standard deviation: $\sqrt{\frac{9(\text{sum of squares}) - (\text{sum})^2}{72}}$

Entry value (mean times the standard deviation):

Name, address:

Recurse: to curse repeatedly

```
10 PRINT "ENTER CURSE AND NUMBER REQUIRED"
20 INPUT C$, N
30 GØSUB 100
40 PRINT "!"
50 END
100 REM RE-CURSING BY RECURSIØN
110 IF N = 0 THEN 150
120 LET N = N - 1
130 GØSUB 100
140 PRINT C$;" ";
150 RETURN
```

```
ENTER CURSE AND NUMBER REQUIRED
? DARN, 5
DARN DARN DARN DARN DARN !
```

```
ENTER CURSE AND NUMBER REQUIRED
? 3 # $ ! @ & * , 4
3 # $ ! @ & * 3 # $ ! @ & * 3 # $ ! @ & * 3 # $ ! @ & * !
```

--To iterate is human, to recurse divine.

--John Motil □

COMPUTER CHESS

by Monroe Newborn, Academic Press, 1975 (ACM Monograph series), 200 pages, hard cover.

Progress in artificial intelligence has been agonizingly slow since the subject became formalized in 1957. Of the subdivisions of the field:

Theorem proving
Music composition
Natural language translation
Game playing
Pattern recognition
Information retrieval
Robotry
Conversational programs
Story and poetry writing

consider how many would show results that would in any way pass Turing's test; namely, that one could not distinguish the output from that produced by a human? There have been signal successes, of course, but the overall picture, for nearly two decades of work and who knows how many millions of dollars, is rather bleak. In the area of pattern recognition, for example, the capability of a house cat is orders of magnitude above that of the best programs. Human capability is orders of magnitude above that of the cat. There is a profound difference between the questions

a. Are these John Brown's fingerprints?

b. Are John Brown's fingerprints in this file?

c. Are John Brown's fingerprints <u>not</u> in this file?

and humans can handle all three, while computer programs are still struggling with the first one.

The most progress, apparently, has been made in the least "useful" areas: music composition and game playing. Acceptable music has been composed by computer programs (the book Music by Computers, edited by Heinz Von Foerster and James Beauchamp, Wiley, 1969, contains four double-sided 7-inch records representing the state of the art as of the date of the book).

The game of Kalah has been demolished; that is, a program for Kalah (a non-trivial open board game) wins the game against human players. Modest capability has been achieved in checkers.

And then there is chess, which has attracted the most attention, the most work, and has shown the most steady progress. This book collects most of the available information on computer programs for chess, up to the 1974 World Computer Chess Championship in Stockholm.

The bulk of the book describes the outstanding programs and gives detailed analyses of games played by those programs. With three exceptions (all from the early days), all the games reported are between two programs, rather than between a program and a competent human player.

Re-play of the 38 games in the book is entertaining and revealing, for the chess novice, the chess expert, and the computist. The best of the programs are still at the amateur level when opposing a Bobby Fischer, but they are constantly improving to close the gap.

The author writes both as a chess expert and a computer expert; he is co-writer of a program called OSTRICH ("because of its cowardly 'head in the sand when in a crisis' style of play"). OSTRICH operates on a Supernova, using 20K words of core, supplemented by a 256K byte disk drive.

The state of the art in chess playing by machine would be better revealed if the playoff of a game between the program and a non-amateur human were given. Still, the book brings together most of what is currently of interest in this small field, without going into details of the programs themselves. The jacket blurb says "it's easy to read," and "the book provides sufficient information to allow someone interested in creating a chess program of his own to get a start in that direction."

SCIENTIFIC ANALYSIS ON THE POCKET CALCULATOR

by Jon M. Smith, John Wiley & Sons, 1975, 380 pages, hard cover.

It has been pointed out that an order of magnitude change (that is, a change by a factor of ten) in physical conditions that affect us will be a change in kind, rather than simply a change in degree. The example usually cited is in transportation, where successive order of magnitude changes were experienced in going from foot travel to horseback, to automobiles, to jet planes; each of these steps led to profound changes in society.

We have already been through at least one full order-of-magnitude change in the performance of pocket calculators per dollar. The earliest machines were simple 4-function devices, selling at $150 in 1971. The equivalent device today (actually better made) sells for $10. The programmable pocket calculator was introduced in 1973 and equivalent capability is now available that offers about five times as much calculator power per dollar. (Consider the SR-52, with a list price of $395 in September 1975: its discount price has gone from $335 March 1, 1976 to $310 April 17.)

Some of this technological and price revolution affects the content of a book like Scientific Analysis on the Pocket Calculator. Smith covers operations on an advanced 4-function machine like the SR-10, a scientific machine like the HP-21/35/45, and a programmable machine like the HP-65. This material (and the lengthy discussion of the relative merits of RPN vs. algebraic notation) is rapidly becoming dated. In particular, the treatment of programmable machines is already quite out of date. The comparison between advanced calculators and true computers (page 286) is very weak and misleading.

But the main thrust of the book--scientific calculations on pocket calculators--is thorough and comprehensive and will remain valid for a long time. Most of the standard topics in Numerical Analysis (function evaluation, series calculations, interpolation, integration, systems of equations) are covered, with the standard techniques modified for application to the pocket machines.

The book includes a wealth of reference material (formulas and algorithms). For most processes, estimates of the expected errors are given. The longest appendix gives algorithms for performing operations in complex arithmetic (up to nice things like complex inverse hyperbolic cosecant), all worked out in terms of the HP-65.

For owners of a simple 4-function machine, there would be little of interest in this book--its topics are too advanced and require use of, at least, a 4-button storage unit. Beginning at the level of the scientific machines (e.g., the SR-50, or HP-21), with serious work to be done, Smith's book is a necessary reference volume.

GAMES WITH THE POCKET CALCULATOR

by Sivasailam Thiagarajan and Harold Stolovitch

Published by Dymax, Box 310, Menlo Park, California 94025, 45 pages 8 1/2 x 11, $2.50.

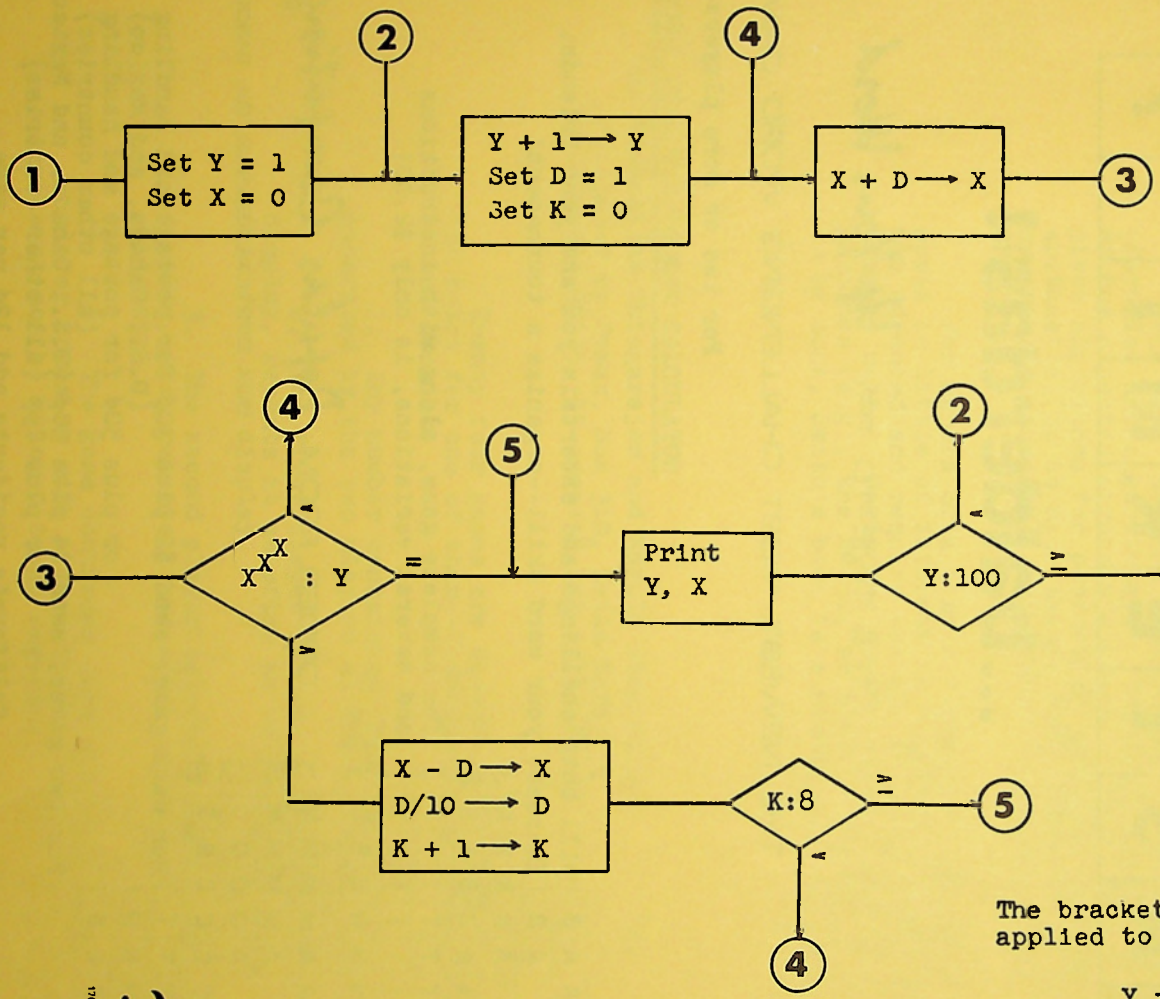
Twenty four games are described in this booklet. The rules for one of them--Countdown--may give the flavor of the booklet:

1. Any number containing three or more digits is generated by the two players as the starting display.

2. The first player selects a single digit in the display, repeats it as often as he pleases, and subtracts this from the display.

3. The second player selects a digit from the new display, also repeats it as often as he wishes, and then subtracts.

4. The game continues back and forth until one player comes up with a zero to win the game. The display must not be allowed to go negative.



The bracketing process applied to the function

$$Y = X^{X^X}$$

PRINTED BY
argold
PRINTING & STATIONERY
17002 Vanowen Blvd., Encino, CA

Bracketing again

In the article on Bracketing (PC35-5) it was pointed out that the process lends itself well to the problem of inverting a function.

Consider the function $Y = X^{X^X}$ with the operations considered in the order $X^{\{X^X\}}$

This function is readily calculated for given values of X:

X	Y
1	1
2	16
3	7.62560×10^{12}
4	1.34078×10^{154}
5	1.91109×10^{2184}

Problem: to calculate a table of values of X for integral values of Y from 2 to 100. A scheme for this calculation is shown in the accompanying flowchart.

Note that X is not reset. The function is monotonously increasing, and hence the last calculated value of X is an excellent starting value for the subsequent calculation.

K is a counter for the digits of precision needed. The limit shown (8) will give the maximum precision for normal Fortran. It is easier to count the number of traverses of the lower leg of the flowchart than to set up elaborate convergence tests.

As long as we are seeking only 100 values of the inverse function, bracketing provides a handy method of solution. The alternatives, such as the Newton-Raphsor scheme, would require taking the derivative of the function:

$$Y' = X^{\left[X^X\right]} + X\left\{\frac{1}{X} + \ln X + (\ln X)^2\right\}$$

which would be quite messy to work with.

KENBAK-1

In POPULAR COMPUTING's second issue (May 1973) the KENBAK-1 computer was described. It went on the market (at around $700) just before the deluge of the micro machines like the Altair 8800 and the build-your-own hobby. The KENBAK-1 is still alive and well, and the price is now nearer $200, fully assembled. It is a 256-word 8-bit machine, with an extensive instruction repetoire. Its creator, John Blankenbaker, describes its most attractive demonstration routine:

HEADS OR TAILS

HEADS OR TAILS is a game played between a person and a computer. Childishly simple, the problem is so complex that a doctoral thesis could be written about it. The requirements to program a solution are almost nil. Simple but bafflingly complex.

What is the game? The human player chooses between binary events which can be called heads and tails. The computer program predicts the player's next choice based on his past history of choices. This prediction is made after the choice but before the next choice is made by the player. A point goes to the computer for a correct prediction and, conversely, the player gets a point for incorrect prediction. In a match of a predetermined number of plays, say 100, the highest score wins.

One strategy for the person would be to generate his choices by a truly random process. This is the best strategy to minimize the number of losses. The challenge to the player is to do better by using his "brain" or by being deliberately deceptive. Thus the proof of superiority for the player consists of winning more than half of the matches.

On the other hand, the computer can win half of its matches by truly random predictions. For the creator of the computer program this is admitting defeat. After all, the sequence of choices made by the player does probably contain some information which would be helpful in prediction. If the program does not win more than half of the time, it is a moral defeat for the programmer.

One possible attempt at a program is to count the number of heads and the number of tails. Predict according to whether more heads or tails choices have been made in the past. Surely the player will have some skew in the number of each choice. By predicting the majority, the program is bound to win, right? WRONG. The score is apt to be 95 for the player and 5 for the program if the player doesn't get bored and quit.

These results arise because the player bases his choices on the prediction made by the program. Most players will discover the program strategy of the previous paragraph within a few moves. They "test" the program, asking how will its predictions be modified by their choices. Perhaps the best summary of the entire interactive process is "I know that he knows that I know...." The writing of a program becomes a non-trivial matter as each side tries to break the code of the other side without disclosing the strategy they are employing.

The author has had fair success with the following type of programs. Keep a score for each of the sixteen possible sequences arising from all combinations of four choices. The prediction is based on the score which corresponds to the last four choices of the player. If the rules for adjusting the score are linear (i.e., adding or subtracting one in accordance with a next choice of heads or tails) the technique does not work too well. The program is sluggish in adapting to changing player strategies and it takes too long to work off the effect of history. By limiting the score to four states (say, strongly heads, weakly heads, weakly tails, and strongly tails) adaptation is faster and performance is better.

When programmed on the KENBAK-1 computer, a hobbyist class of machine, about 64 bytes of memory are required. As a demonstration program it is effective, being easy to understand and play. Most of the fun arises from the programming. Will this version or refinement work better? The author's experience is that the programmer does not make a good player, perhaps because his interest is divided between wanting to win as a player and wanting to win as a programmer. So enlist your friends and family as opponents. □

Log 39	1.591064607026499206501533061197443740029858052577764
ln 39	3.663561646129646427448732678487844309452758502582957
$\sqrt{39}$	6.244997998398398205846893120939794461072959977991656
$\sqrt[3]{39}$	3.391211443014166761954670725994596733262836124809925
$\sqrt[10]{39}$	1.442468907554628620631033606028189152472577084137390
$\sqrt[100]{39}$	1.037314971464803308101025911299167256641947217431821
e^{39}	86593400423993746.953606932719264934249701854700195986591516529602527699530004594190243 1
π^{39}	24481895231475088054.641675209164399276378300198876017867794796831879079305033793348 0
$\tan^{-1}$ 39	1.545160918273219138538622041219857545054815298232215

Four things should always be remembered:

1. Edit the incoming data

2. Patch the disease, not the symptoms

3. Seven times eight isn't sixty three

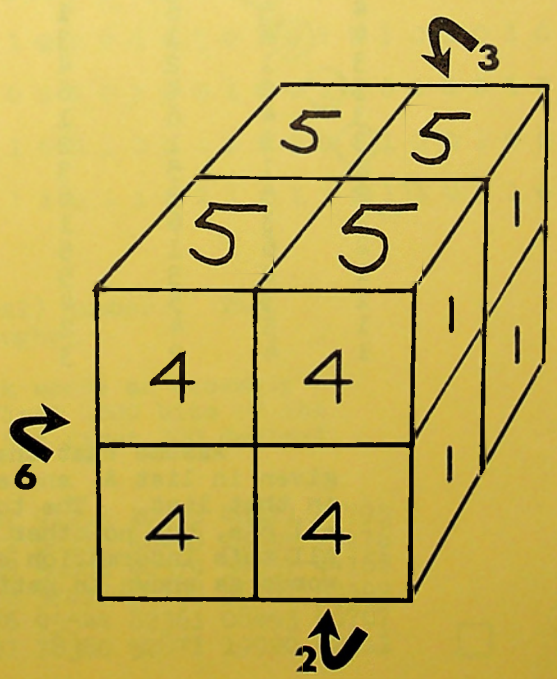
8 Dice

Given eight dice, numbered on their faces as follows:

A

die number	top	front	left	right	back	bottom
1	6	3	2	4	5	1
2	6	4	2	1	5	3
3	6	4	2	1	5	3
4	6	3	1	4	5	2
5	6	3	2	5	1	4
6	6	3	5	4	1	2
7	6	3	1	2	5	4
8	6	3	1	5	4	2

The eight small cubes can be arranged in the form of a 2 x 2 x 2 cube:



(Note that none of these arrangements follows that of a commercial die, which has the order 6,3,5,2,4,1, with opposing faces adding to 7.)

in which the numbers on the external faces are in the order of a commercial die, and this can be done in essentially only one way. As with the Hexagon Problem (our Problem No. 9, in issue No. 5) there are an enormous number of arrangements that are possible, but not all of them need be considered in the process of finding the desired arrangment.

There are 8! = 40320 permutations of the spatial positions of the dice. For each of these permutations, each of the dice may have to be rotated in place through 24 positions. For the first die, these 24 positions are as shown in this table:

top	front	left	right	back	bottom
6	3	2	4	5	1
6	2	4	5	3	1
6	4	5	3	2	1
6	5	3	2	4	1
5	1	4	2	6	3
5	4	2	6	1	3
5	2	6	1	4	3
5	6	1	4	2	3
4	1	3	5	6	2
4	3	5	6	1	2
4	5	6	1	3	2
4	6	1	3	5	2
3	1	2	4	6	5
3	2	4	6	1	5
3	4	6	1	2	5
3	6	1	2	4	5
2	1	5	3	6	4
2	5	3	6	1	4
2	3	6	1	5	4
2	6	1	5	3	4
1	2	3	5	4	6
1	3	5	4	2	6
1	5	4	2	3	6
1	4	2	3	5	6

Assume that the dice are arranged in the order given in list A, and each die is in the orientation given in that list. The top of the large cube will then have all 6's, but no other face will have all alike digits. All this information can readily be stored in 48 computer words as shown in pattern B:

	top	front	left	right	back	bottom
1	(6)	(3)	(2)	4	5	1
2	(6)	(4)	2	(1)	5	3
3	(6)	4	2	(1)	(5)	3
4	(6)	3	(1)	4	(5)	2
5	6	(3)	(2)	5	1	(4)
6	6	3	(5)	4	(1)	(2)
7	6	3	1	(2)	(5)	(4)
8	6	(3)	1	(5)	4	(2)

B

The circled numbers in each column must be the same, and the numbers from 1 to 6 must appear in sets of 4 in the 6 columns. If the 8 dice are thought of as counter wheels (each wheel having 24 positions), then the test can be performed on the wheels in some systematic order. For example, in pattern B, the conditions for the top face of the large cube are satisfied. Die number 2 can now be rotated, but through only 4 of its 24 positions, since the 6 must remain on top. Since the 3 of die number 1 is in front, we can attempt to rotate die number 2 to conform, and this is quickly found to be impossible. There is no point in attempting to rotate dice numbers 3 to 8; a new arrangement should be tried immediately. In short, we have another nice problem in Backtracking.

(A) Draw a flowchart for the logic of manipulating the cubes as numbers in storage, and testing each new arrangement to find the one that solves the problem.

(B) Write a program to implement that flowchart.

(C) Devise a procedure to test the program before committing it to production.

The problem is similar to Exercise 15 in the book Introduction to Fortran, by Sten Kallin. The problem was given there with cubes having six different colors on their faces. The exercise has this note:

Eight cubes may be arranged in

$$8! \cdot 24^7 = 1.85 \cdot 10^{14}$$

ways. There would be small reward in allowing the machine to try out all these alternatives: the cost of execution time would be most unreasonable.

PROBLEM 132

The following table gives the products of groups of 100 consecutive integers. The first entry is 100! The second entry is the product of (101)(102)(103)···(200), which is (200!)/(100!). The last entry is (4100!)/(4000!).

For each entry, the first 50 significant digits are given, plus the number of digits to the decimal point. Thus, the last entry shows the first 50 digits of a 361-digit number.

The table was calculated in 1964 by Harold Thompson on an IBM 1620.

Centafactorials

100	93326215443944152681699238856266700490715968264381	108
200	84505501869246294958381570938554045654413667220124	167
300	38807387193016483645683371924167275439580023008808	190
400	20922382327063861735861920196483244905990662825187	205
500	19054359613385322234269048956226032795163441871435	216
600	10372380287738743469071640620604012340151668001286	225
700	19137903806034620887669233729171302862705021888059	232
800	31834857047006456950844560140635160296819069457706	238
900	87577379527636153943566695560374055358478694434350	243
1000	59589266322404781554893890579461327225982795887772	248
1100	13280014101512138589141872492770391536502915448947	253
1200	11884609911818940553649199588783759729381959177233	257
1300	49750022854788016838101815448479277550211544344112	260
1400	10953025871149212040000082562417606497113181746918	264
1500	13904978828840919544040221606929549270013338224218	267
1600	10955906808478114573924138962664581795866688886882	270
1700	56873380367728839514457286459557267632311508940193	272
1800	20431735684772603813768050198705961187290399021266	275
1900	52920498171432627023990335783260621301590523490874	277
2000	10229127037233393827003242294009996867186071204316	280
2100	15194038089234765044041792663716751665621792781228	282
2200	17784087520549485529184791553708601273318346723036	284
2300	16762110667316765935820071701059236261540278349501	286
2400	12964331404306686126482751131532307812202741136457	288
2500	83648308295111829000436400755199554366486315959681	289
2600	45682616093443881292967439699902244094763088218787	291
2700	21389715461146884229881130015062357745817943822307	293
2800	86850634079710094923956703904289787157644234845797	294
2900	30894238576812401179839926946319459705223826909123	296
3000	97159053219694503084196412031669503387151930873477	297
3100	27236943337809709587270029672198942249720242936087	299
3200	68568606662162772218666299774275746139893795362409	300
3300	15606460628037577755442872001184884194093978905267	302
3400	32310994274131996932992826550283587179305685542380	303
3500	61190408358107137282993168123583992934418980269630	304
3600	10654088179026667976632840118654307217895044206664	306
3700	17134736700459134333189759313626754491036705695833	307
3800	25564104521681073766809021682891870780811579750763	308
3900	35521584513789755420595305138420300228341949882235	309
4000	46136769119179142839059001473604691292375542972942	310
4100	56203163425678075165724011194795228526295867104766	311